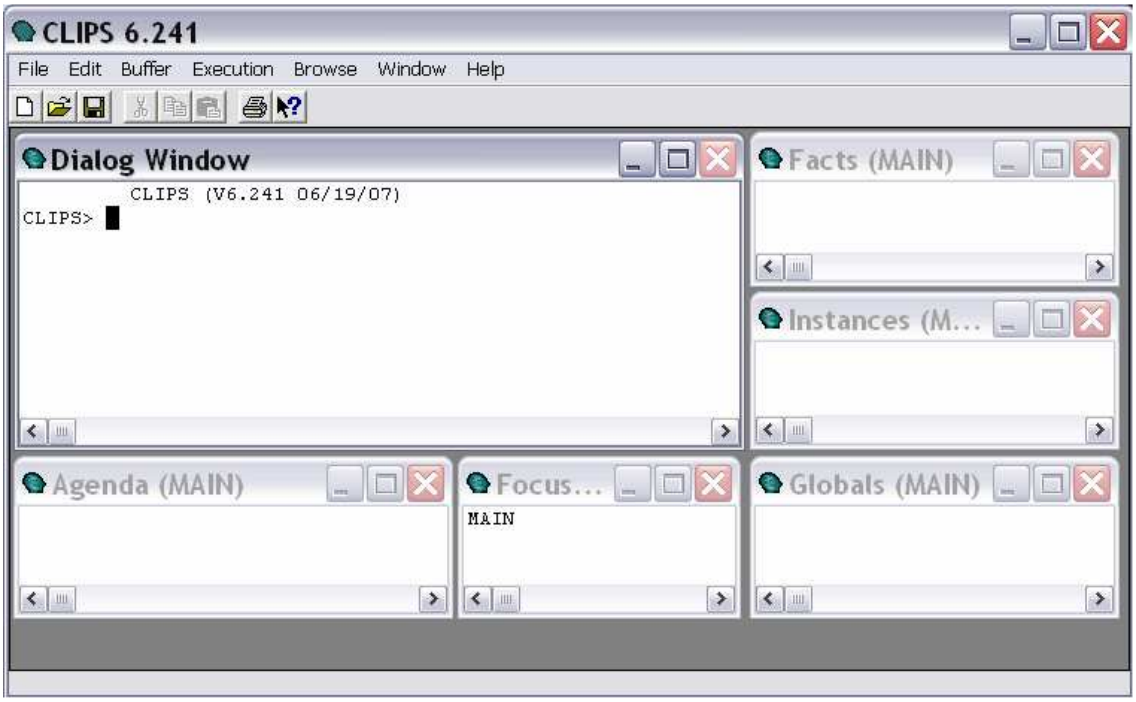
ΑΣΚΗΣΗ 1

Το εργαλείο CLIPS είναι ένα περιβάλλον κατάλληλο για προγραμματισμό με κανόνες, συναρτήσεις και αντικείμενα. Στόχος της πρώτης εργαστηριακής άσκησης είναι η εξοικείωση με το περιβάλλον αυτό μέσω της χρήσης κάποιων απλών προγραμμάτων.

Το CLIPS εκκινεί σε περιβάλλον MS Windows με την εκτέλεση του αρχείου **CLIPSWin.exe**, το οποίο μπορείτε να βρείτε και να κατεβάσετε στην παρακάτω διεύθυνση (επιλέγετε **clipswin_executable_6241.zip**):

http://clipsrules.sourceforge.net/

Μετά την εκτέλεση του CLIPSWin.exe εμφανίζεται το γραφικό περιβάλλον της γλώσσας:



Στην αρχή είναι ανοιχτό μόνο το Παράθυρο Διαλόγου (**Dialog Window**) αλλά μπορούμε να ανοίξουμε και τα **Status Windows** μέσα από το μενού Window.

Αλληλεπίδραση με το CLIPS

Όπως αναφέρεται και στις σημειώσεις του μαθήματος, η σύνταξη της γλώσσας του CLIPS θυμίζει σε αρκετά σημεία την σύνταξη της LISP, την οποία έχετε διδαχθεί. Η **κατάλληλη χρήση παρενθέσεων** είναι κι εδώ απαραίτητη όταν δίνονται εντολές. Αφού ανοίξει το Παράθυρο Διαλόγου και εμφανιστεί το prompt:

CLIPS>

μπορούμε να ξεκινήσουμε να δίνουμε άμεσα εντολές στο CLIPS (**top level commands**).

- Πληκτρολογήστε τις παρακάτω εντολές στο περιβάλλον CLIPS και σημειώστε τα αποτελέσματα που εκτυπώνονται:

1. (+ 3 4)
2. (printout t “I am bored!”)
3. (defglobal ?*x* = 3)
4. ?*x*
5. (>= 7 7 5 3 1)
6. +12
7. (deftemplate person (slot name) (slot age))
8. -32.3e-7
9. (one two three)
10. (create$ one two three)
11. (or (> 2 3) (< 5 4))
12. Hello
13. (printout t “I am ” crlf “still bored!” crlf)
14. (+ 3 (* 8 9) 4)
15. (and (bind ?x 10) (printout t “The next of “ ?x “is ” (+ ?x 1) crlf)
16. (- ?*x* 2)
17. ?*x*
18. (eq foo Foo)
19. (symbolp @+=-%)
20. (* 5 6.0 2)
21. "a and b"
22. (floatp 3)
23. (* 8 + 3 (* 2 3 4) 9) * 3 4))
24. (assert (person Panagiotis 24 musician))

25. (stringp abcd)

26. (symbolp “abcd”)

27. (bind ?y 30)

28. (facts)

- Όπως παρατηρούμε, μια εντολή στο CLIPS μπορεί να είναι είτε **κλήση κάποιας συνάρτησης**, είτε **ορισμός κάποιας δομής**, είτε κάποια **καθολική μεταβλητή** είτε, τέλος, κάποια **σταθερά**. Όταν καλείται μια συνάρτηση εκτιμάται η επιστρεφόμενη τιμή της (return value) και εκτυπώνεται. Ο ορισμός δομής έχει σαν αποτέλεσμα την δημιουργία μιας δομής κάποιου συγκεκριμένου τύπου. Όταν εισάγεται μετά το prompt μια καθολική μεταβλητή εκτυπώνεται η τιμή της ενώ όταν εισάγεται σαν top-level εντολή μια σταθερά εκτυπώνεται η ίδια η σταθερά (κάτι που δεν έχει ιδιαίτερη χρησιμότητα). Με βάση αυτά:

    1. Ποιες από τις προηγούμενες εντολές αποτελούν κλήσεις συναρτήσεων; Για καθεμία σημειώστε τον αριθμό της.

    2. Ποιες από τις προηγούμενες εντολές αποτελούν ορισμούς δομής; Για καθεμία σημειώστε τον αριθμό της.

    3. Ποιες από τις προηγούμενες εντολές αποτελούν καθολικές μεταβλητές; Για καθεμία σημειώστε τον αριθμό της.

    4. Ποιες από τις προηγούμενες εντολές αποτελούν σταθερές; Για καθεμία σημειώστε τον αριθμό της.

    5. Για κάθε απάντηση στο 1, βρείτε τον τύπο της αντίστοιχης συνάρτησης. Οι πιθανές κατηγορίες συναρτήσεων του συστήματος είναι οι εξής: **αριθμητική**, **σύγκρισης**, **λογική**, **ελέγχου τύπου**, **χειρισμού πολλαπλών τιμών**, **εισόδου**, **εξόδου**, **ανάθεσης**, **ελέγχου ροής** και **ελέγχου περιβάλλοντος**. Τι παρατηρείτε σχετικά με την σύνταξη των συναρτήσεων.

    6. Για κάθε απάντηση στο 4, βρείτε τον τύπο δεδομένων της αντίστοιχης σταθεράς.

    7. Σχολιάστε το αποτέλεσμα της εντολής 9.

    8. Υπάρχουν λανθασμένες εντολές (εκτός από την 9) στο σύνολο εντολών της προηγούμενης άσκησης; Αν ναι, ποιες είναι και γιατί είναι λανθασμένες; Προσθέστε ότι λείπει σε καθεμία για να διορθωθεί. Στη συνέχεια, σημειώστε το αποτέλεσμα της διορθωμένης εντολής.

    9. Πού χρησιμοποιείται το ‘crlf’ και ποιος είναι ο ρόλος του; Τι σημαίνει το ‘t’;

10. Σχολιάστε το αποτέλεσμα της ακολουθίας εντολών 16,17. Ποια εντολή θα έπρεπε να δώσουμε για να αλλάξουμε την τιμή του ?*x*;

11. Ανοίξτε το Status Window **Facts** από το μενού Window με την επιλογή **Facts** (για να ανοίξουν ταυτόχρονα όλα τα status windows επιλέγουμε **Show Status Windows** αλλά για την ώρα δεν μας απασχολούν τα υπόλοιπα παράθυρα). Σχολιάστε το περιεχόμενό του. Ποια εντολή ευθύνεται για αυτό; Τι ρόλο παίζει το παράθυρο Facts; Εκτελέστε τώρα την εντολή: (retract 0). Σχολιάστε το όρισμα που δέχεται και το αποτέλεσμά της.

- Με την χρήση της εντολής περιβάλλοντος **(clear-window)** ή με την επιλογή **Clear Dialog Window** από το μενού Window, καθαρίστε το Παράθυρο Διαλόγου από τις top-level εντολές που εκτελέσατε ως τώρα. Με χρήση της συνάρτησης ανάθεσης bind δώστε σε μια μεταβλητή ?x την τιμή 20. Στην συνέχεια, δοκιμάστε να εκτυπώσετε την τιμή της ?x ως εξής:

  (printout t "The value of the variable ?x is " ?x crlf)

  Τι συμβαίνει; Εξηγείστε το μήνυμα που εμφανίζεται. Για να καταλάβετε καλύτερα γράψτε την παρακάτω εντολή και παρατηρήστε το αποτέλεσμα:

  (and (bind ?x 20) (printout t "The value of the variable ?x is " ?x crlf))

  Ποια είναι η διαφορά τώρα (στην οποία οφείλεται η επιτυχημένη εκτύπωση); *(Παρατηρήστε ότι εδώ η συνάρτηση 'and' προκαλεί την εκτέλεση των δύο εντολών-ορισμάτων της προκειμένου να ικανοποιήσει την λογική λειτουργία της)*

- Εισάγετε στο CLIPS τα παρακάτω σύμβολα και σχολιάστε τα αποτελέσματα:

  1. JsQs<sld
  2. 3JsQs>sld
  3. me&you
  4. <x1000
  5. 23+25
  6. k$s
  7. –etc
  8. nice(job

  Ποια από τα παραπάνω θεωρούνται σύμβολα και ποια όχι; Σημειώστε τους χαρακτήρες που ευθύνονται για τα λανθασμένα σύμβολα. Ποιο είναι το αποτέλεσμα αυτών των χαρακτήρων; Υπάρχει κάποια εξαίρεση; Τι περιμένετε να εκτυπώσει το CLIPS όταν εισάγετε τα παρακάτω και γιατί;

  CLIPS> one two three

- Ποιο περιμένετε να είναι το αποτέλεσμα των παρακάτω συναρτήσεων χειρισμού πολλαπλών τιμών και ελέγχου τύπου:

  1. (create$ friends bill elina)
  2. (create$ a (create$ 2 3) b (create$ 4 5) c)
  3. (member$ 2 (1 2 3))
  4. (member$ abc (create$ “abc” 2 4 5))
  5. (member$ “a” (create$ a b ac “a”))
  6. (symbolp (first$ (create$ o p q r)))
  7. (first$ (rest$ (create$ o p q r)))
  8. (eq (nth$ 1 (create$ a b c)) (first$ (create$ a b c)))
  9. (symbolp (nth$ 1 (create$ a b c)))

  Εκτελέστε τις παραπάνω εντολές στο CLIPS και δείτε αν επαληθεύονται οι απαντήσεις που δώσατε. Αν κάποια είναι λάθος διορθώστε την. Με βάση τα αποτελέσματά τους περιγράψτε την λειτουργία καθεμιάς από τις συναρτήσεις που εμφανίζονται εδώ.

- Μετατρέψτε την λίστα (I don’t have a name) σε string χρησιμοποιώντας κατάλληλη εντολή του CLIPS. Στη συνέχεια κάνετε το αντίστροφο, δηλαδή μετατρέψτε το string “I don’t have a name” σε λίστα. Γράψτε μια εντολή που θα έχει σαν αποτέλεσμα την εκτύπωση του παρακάτω:

  ”I‘m fine!” the man said.
  “Don’t worry”

- Τώρα θα δούμε ποια είναι η λειτουργία των παραθύρων **Facts** και **Agenda** που αντιστοιχούν στην μνήμη εργασίας και στην στοίβα κανόνων, αντίστοιχα (*βλ. σημειώσεις μαθήματος σελ.92-93*). Με την επιλογή **File->New** δημιουργείστε ένα νέο αρχείο και αντιγράψτε τους δύο παρακάτω ορισμούς σε αυτό:

```
(deffacts interestes
   (interested popi in tango)
   (interested john in music)
   (interested andronikos in music)
   (interested andronikos in tango))

(defrule print-matches
   ?fid1 <- (interested ?sb in ?something)
   ?fid2 <- (interested ?sb2 in ?something)
=>
   (if (neq ?sb ?sb2) then
      (assert (matches ?sb ?sb2))
      (retract ?fid1)
      (retract ?fid2)))
```

  Σώστε το αρχείο με όποιο όνομα θέλετε αλλά χωρίς να ξεχάσετε να βάλετε την σωστή επέκταση: **.clp**.

Στο αρχείο που μόλις δημιουργήσαμε ορίζεται μια λίστα από γεγονότα που περιγράφουν την αρχική κατάσταση του προβλήματός μας. Κάθε γεγονός δηλώνει πως κάποιο άτομο ενδιαφέρεται για κάτι. Στη συνέχεια ορίζεται ένας κανόνας ο οποίος με βάση τα ενδιαφέροντα κάποιων ατόμων βρίσκει ποια από αυτά ταιριάζουν μεταξύ τους και εισάγει στην μνήμη την νέα πληροφορία με την μορφή γεγονότων της μορφής (matches <άτομο1> <ατομο2>) διαγράφοντας παράλληλα την πληροφορία (γεγονότα) που χρησιμοποίησε και που δεν μας χρειάζεται πια.

Με τα γεγονότα και τους κανόνες θα ασχοληθούμε εκτενέστερα στις επόμενες εργαστηριακές ασκήσεις, οπότε αυτή τη στιγμή δεν μας απασχολεί και τόσο η σύνταξη των παραπάνω ορισμών. Αυτό που μας ενδιαφέρει εδώ είναι η γενική διαδικασία που ακολουθούμε για να δημιουργήσουμε και να εκτελέσουμε ένα αρχείο με κώδικα CLIPS, καθώς και ο ρόλος των δύο βασικότερων status windows.

Αφού, λοιπόν, δημιουργήσαμε και σώσαμε το αρχείο μας πρέπει ό,τι ορίσαμε μέσα σε αυτό να φορτωθεί με κάποιον τρόπο στα κατάλληλα μέρη του CLIPS. Θυμίζουμε την βασική δομή του CLIPS:

- Λίστα Γεγονότων (Facts)
- Βάση Κανόνων
- Στοίβα Ενεργοποιημένων Κανόνων (Agenda)
- Μηχανισμός Εξαγωγής Συμπερασμάτων.

Ανοίγουμε από το μενού Window τα status windows **Facts** και **Agenda**. Χρησιμοποιούμε την επιλογή **File->Load** για να φορτώσουμε το αρχείο μας.

1. Τι αλλαγές παρατηρείτε στα windows Facts και Agenda μετά την φόρτωση του αρχείου; Τι θα έπρεπε να συμβαίνει;

2. Εισάγετε την εντολή **(reset)** μετά το prompt στο Dialog Window ή επιλέξτε **Execution->Reset**. Τι παρατηρείτε; Τι κάνει η εντολή reset; Πληκτρολογήστε την εντολή (facts) και δείτε τι κάνει.

3. Μετά το (reset) και την καταχώρηση των γεγονότων που ορίσαμε στην μνήμη εργασίας, παρατηρούμε μια αλλαγή και στο παράθυρο Agenda. Πού οφείλεται αυτή η αλλαγή; Προσπαθήστε να εξηγήσετε τα περιεχόμενα της ατζέντας. Παίζει κάποιο ρόλο η σειρά τους και αν ναι ποιόν;

4. Ένας κανόνας ενεργοποιείται όταν ικανοποιούνται όλες οι συνθήκες του. Η ενεργοποίηση ενός κανόνα αρκεί για να οδηγήσει στην εκτέλεσή του; Αν όχι, ποια είναι η εντολή που χρειάζεται ώστε να επιτραπεί στους ενεργοποιημένους κανόνες να εκτελεστούν;

5. Προσπαθήστε να φανταστείτε (με βάση την λειτουργία του κανόνα που ορίσαμε) ποιο θα είναι το αποτέλεσμα της εντολής **(run)**. Στη συνέχεια γράψτε την μετά το prompt ή επιλέξτε **Execution->Run** και παρατηρήστε τι συμβαίνει.

6. Δώστε ξανά την εντολή (reset). Τι γίνεται; Στην συνέχεια δώστε την εντολή **(clear)**. Ποια είναι η διαφορά της εντολής (clear) από την (reset); Παρατηρήστε πως αν μετά την εντολή (clear) ξαναδώσετε την εντολή (reset) δεν θα ξαναφορτωθούν στην μνήμη τα προηγούμενα γεγονότα. Για να ξαναγίνει αυτό τώρα πρέπει να ξαναφορτώσουμε το αρχείο που τα περιέχει.

- Καθαρίστε ξανά το Παράθυρο Διαλόγου και με την εντολή περιβάλλοντος **(clear)** σβήστε κανόνες και γεγονότα από την μνήμη. Επιλέξτε **Turn Dribble On** από το μενού **File**. Δημιουργείται ένα αρχείο dribble.txt που τοποθετείται σε όποιον φάκελο θέλετε. Στην συνέχεια εκτελέστε τις παρακάτω εντολές και παρατηρήστε τι κάνουν:

  1. (subsetp (create$ eleni tasos) (create$ bill eleni tasos kate))

  2. (delete$ (create$ hammer drill saw pliers wrench) 3 5))

  3. (replace$ (create$ drill wrench pliers) 3 3 machete)

  4. (length$ (create$ a b c d e f g))

  Στη συνέχεια, επιλέξτε **Turn Dribble Off** από το μενού **File**. Ανοίξτε το αρχείο dribble.txt και παρατηρήστε την λειτουργία της συγκεκριμένης επιλογής του μενού File.

- Κλείστε το CLIPS είτε με την εντολή **(exit)** στο Dialog Window είτε με **File->Exit**.